Forking

Unix is a multi-programming system. The kernel brings the system up at booting and then it stops leaving its children, namely tasks and drivers, to run. All user processes are children of `init`.

The only way to create a new process in Unix is by using the `fork` command. This command is declared in the *unistd* header.

Example 1 demonstrates how `fork` works. Here the `if–else` conditional statement has two possibilities both of which run in parallel to each other. The *stdlib* header is there because `system` needs it, the *stdio* because `printf` does.

Example 1. (Example of forking and child processes)

```
1  /* two children running in parallel, Kit Tyabandha, 30 Jan 07 */
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <unistd.h>
5  int
6  main(){
7    printf("\n Parent\n\n");
8    if(fork()!=0){
9      printf(" Child 1\n\n");
10     system("date");
11   } else {
12     printf(" Child 2\n\n");
13     system("whoami");
14   }
15   return 0;
16 }
```

Bibliography

Andrew S Tanenbaum and Albert S Woodhall. *Operating systems design and implementation.* 3rd ed., Prentice-Hall, 2006(1987)